(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 1 378 821 A2

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
07.01.2004 Bulletin 2004/02

(51) Int Cl.[7]: **G06F 7/72**, H04L 9/32

(21) Application number: **03254190.6**

(22) Date of filing: **01.07.2003**

(84) Designated Contracting States:
AT BE BG CH CY CZ DE DK EE ES FI FR GB GR HU IE IT LI LU MC NL PT RO SE SI SK TR
Designated Extension States:
AL LT LV MK

(30) Priority: **05.07.2002 GB 0215590**

(71) Applicant: **Hewlett-Packard Development Company, L.P.**
Houston, Texas 77070 (US)

(72) Inventors:
- **Chen, Liqun**
  Bradley Stoke, Bristol BS32 9DQ (GB)
- **Harrison, Keith Alexander**
  Chepstow, Monmouthshire NP16 7PX (GB)
- **Soldera, David**
  Horfield, Bristol BS7 8PQ (GB)

(74) Representative: **Squibbs, Robert Francis et al**
Hewlett-Packard Limited,
IP Section,
Building 3
Filton Road
Stoke Gifford, Bristol BS 34 8QZ (GB)

(54) Authentication method and apparatus using pairing functions for the elliptic curves based cryptosystems

(57) A first party (60) has a first (s_1) and a second (P,R_{TA1}) cryptographic key. A second party (70) has a third ("TA2") and a fourth ($s_1 Q_{TA2}$) cryptographic key, the fourth cryptographic key ($s_1 Q_{TA2}$) being derived from the first (s_1) and third ("TA2") cryptographic keys thereby providing an association between the parties (60,70). To enable a third party (90) to verify the existence of an association between the first and second parties (60,70), the second party generates a number (r) that in association with the second cryptographic key (P,R_{TA1}), the third cryptographic key ("TA2") and the fourth ($s_1 Q_{TA2}$) cryptographic key define a first cryptographic parameter (X), a second cryptographic parameter (Y) and a third cryptographic parameter (Z) respectively. By using these parameters and the second and third cryptographic keys, the third party (90) can verify if the first and second parties (60,70) are associated.

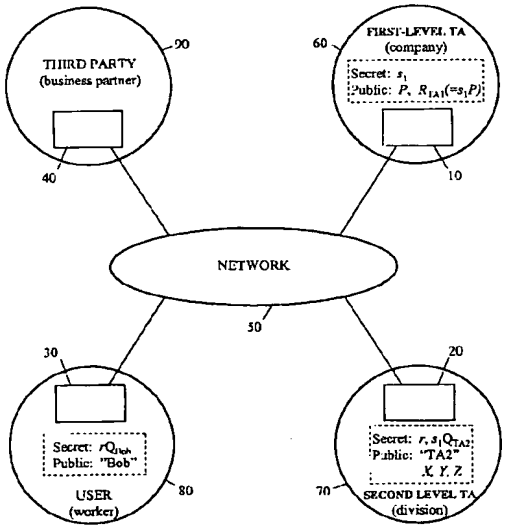

Figure 3

EP 1 378 821 A2

Description

Field of the Invention

[0001] The present invention relates to a method and apparatus for use relation to verifying an association between two parties by cryptographic techniques; in particular, but not exclusively, the present invention relates to a method and apparatus for enabling the verification, and/or for verifying, an association between a lower-level trusted authority and a higher-level trusted authority in a hierarchy of trusted authorities by using elliptic curve cryptography.

Background of the Invention

[0002] With the ever-increasing spread of electronic communication and electronic identification there has been a corresponding increase in demand for cryptographic processes, where users require cryptographic processes to enable encryption of data for security purposes and/or for the purposes of providing identification.

[0003] Typically encryption keys are certified by trusted authorises and disseminated using digital certificates where, to allow wide spread availability of cryptographic processes, a hierarchy of trusted authorities exist. Within a hierarchy of trusted authorities a root trusted authority issues a digital certificate to a private/public key to a second level trusted authority by using the root authorities private key to sign the second level's trusted authorities public key and thereby providing confirmation that the second level private key is authorized by the root authority. Correspondingly the second level trusted authority issues a digital certificate to a different private/public key to a third level trusted authority that is signed with the second level's private key and so forth. However, for a user to determine that the public key associated with the third level trusted authority is derived with the authority of the root trusted authority it is necessary for the user to trace the digital certificates that incorporated the various public keys.

[0004] It is desirable to improve this situation.

[0005] Embodiments of the present invention to be described hereinafter make use of cryptographic techniques using bilinear mappings. Accordingly, a brief description will now be given of certain such prior art techniques.

[0006] In the present specification, G_1 and G_2 denote two algebraic groups of prime order q in which the discrete logarithm problem is believed to be hard and for which there exists a computable bilinear map $\hat{p}$, for example, a Tate pairing t or Weil pairing $\hat{e}$. Thus, for the Weil pairing:

$$\hat{e}: G_1 \times G_1 \rightarrow G_2$$

where G_2 is a subgroup of a multiplicative group of a finite field. The Tate pairing can be similarly expressed though it is possible for it to be of asymmetric form:

$$t: G_0 \times G_1 \rightarrow G_2$$

where G_0 is a further algebraic group the elements of which are not restricted to being of order q. Generally, the elements of the groups G_0 and G_1 are points on an elliptic curve though this is not necessarily the case.

[0007] As is well known to persons skilled in the art, for cryptographic purposes, a modified form of the Weil pairing is used that ensure $\hat{p}(P,P) \neq 1$ where $P \in G_1$; however, for convenience, the pairing is referred to below simply by its usual name without labeling it as modified. Further background regarding Weil and Tate pairings and their cryptographic uses can be found in the following references:

- G. Frey, M. Müller, and H. Rück. The Tate pairing and the discrete logarithm applied to elliptic curve cryptosystems. *IEEE Transactions on Information Theory*, 45(5):1717-1719, 1999.
- D. Boneh and M. Franklin. Identity based encryption from the Weil pairing. In *Advances in Cryptology - CRYPTO 2001*, LNCS 2139, pp. 213-229, Springer-Verlag, 2001.

[0008] For convenience, the examples given below assume the use of a symmetric bilinear map ($\hat{p}: G_1 \times G_1 \rightarrow G_2$) with the elements of G_1 being points on an elliptic curve; however, these particularities, are not to be taken as limitations on the scope of the present invention.

[0009] As the mapping between G_1 and G_2 is bilinear exponents/multipliers can be moved around. For example if $a, b, c \in F_q$ and $P, Q \in G_1$ then

$$t(aP, bQ)^c = t(aP, cQ)^b = t(bP, cQ)^a = t(bP, aQ)^c = t(cP, aQ)^b = t(cP, bQ)^a$$
$$= t(abP, Q)^c = t(abP, cQ) = t(P, abQ)^c = t(cP, abQ)$$
$$= \ldots$$
$$= t(abcP, Q) = t(P, abcQ) = t(P, Q)^{abc}$$

[0010] Additionally, the following cryptographic hash functions are defined:

$$H_1 : \{0,1\}^* \rightarrow G_1$$

$$H_2 : \{0,1\}^* \rightarrow F_q$$

$$H_3 : G_2 \rightarrow \{0,1\}^*$$

[0011] A normal public/private key pair can be defined for a trusted authority:

the private key is s where $s \in F_q$
the public key is (P, R) where $P \in G_1$ and $R \in$ G1, with $R=sP$

[0012] Additionally, an identifier based public key / private key pair can be defined for a party with the cooperation of the trusted authority. As is well known to persons skilled in the art, in "identifier-based" cryptographic methods a public, cryptographically unconstrained, string is used in conjunction with public data of a trusted authority to carry out tasks such as data encryption or signing. The complementary tasks, such as decryption and signature verification, require the involvement ofthe trusted authority to carry out computation based on the public string and its own private data. Frequently, the string serves to "identify" the intended message recipient and this has given rise to the use of the label "identifier-based" or "identity-based" generally for these cryptographic methods. However, depending on the application to which such a cryptographic method is put, the string may serve a different purpose to that of identifying the intended recipient and, indeed, may be an arbitrary string having no other purpose than to form the basis of the cryptographic processes. Accordingly, the use of the term "identifier-based" herein in relation to cryptographic methods and systems is to be understood simply as implying that the methods and systems are based on the use of a cryptographically unconstrained string whether or not the string serves to identify the intended recipient. Furthermore, as used herein the term "string" is simply intended to imply an ordered series of bits whether derived from a character string, a serialized image bit map, a digitized sound signal, or any other data source.

[0013] In the present case, the identifier-based public / private key pair defined for the party *has a public key* Q_{ID} and private key S_{ID} where Q_{ID}, $S_{ID} \in G_1$. The trusted authority's normal public/private key pair $(P,R/s)$ is linked with the identifier-based public/private key by

$$S_{ID} = sQ_{ID} \text{ and } Q_{ID} = H_1 (ID)$$

where ID is the identifier string for the party.

[0014] Some typical uses for the above described key pairs will now be given with reference to Figure 1 of the accompanying drawings that depicts a trusted authority 10 with a public key *(P, sP)* and a private key *s*. A party A serves as a general third party whilst for the identifier-based cryptographic tasks (IBC) described, a party B has an IBC public key Q_{ID} and an IBC private key S_{ID}.

[0015] Standard Signatures *(see dashed box 2):* The holder of the private key s (that is, the trusted authority 1 or anyone to whom the latter has disclosed *s*) can uses to sign a bit string; more particularly, where *m* denotes a message to be signed, the holder of *s* computes:

$$V = sH_1(m).$$

[0016] Verification by party A involves this party checking that the following equation is satisfied:

$$t(P,V)=t(R,H_1(m))$$

[0017] This is based upon the mapping between G_1 and G_2 being bilinear exponents/multipliers, as described above. That is to say,

$$\begin{aligned} t(P, V) &= t(P, sH_1(m)) \\ &= t(P, H_1(m))^s \\ &= t(sP, H_1(m)) \\ &= t(R, H_1(m)) \end{aligned}$$

[0018] Identifier-Based Encryption *(see dashed box 3)*: - Identifier based encryption allows the holder of the private key S_{ID} of an identifier based key pair (in this case, party B) to decrypt a message sent to them encrypted (by party A) using B's public key Q_{ID}.

[0019] More particularly, party A, in order to encrypt a message m, first computes:

$$U = rP$$

where r is a random element of F_q. Next, party A computes:

$$V = m \oplus H_3(t(R, rQ_{ID}))$$

[0020] Party A now has the ciphertext elements U and V which it sends to party B.

[0021] Decryption of the message by party B is performed by computing:

$$\begin{aligned} V \oplus H3(t(U, S_{ID})) &= V \oplus H_3(t(rP, sQ_{ID})) \\ &= V \oplus H_3(t(P, Q_{ID})^{rs}) \\ &= V \oplus H_3(t(sP, rQ_{ID})) \\ &= V \oplus H_3(t(R, rQ_{ID})) \\ &= m \end{aligned}$$

[0022] Identifier-Based Signatures *(see dashed box 4)* : - Identifier based signatures using Tate pairing can be implemented. For example:

Party B first computes:

$$r = t(S_{ID}, P)^k$$

where k is a random element of F_q.

Party B then apply the hash function H_2 to $m \,||\, r$ (concatenation of m and r) to obtain:

$$h = H_2(m \| r).$$

[0023] Thereafter party B computes

$$U = (k\text{-}h)S_{ID}$$

thus generating the output *U and h* as the signature on the message *m.*

[0024] Verification of the signature by party A can be established by computing:

$$r' = t(U, P) \bullet t(Q_{ID}, R)^h$$

where the signature can only be accepted if $h = H_2 (m \| r')$.

[0025] It will be recalled that the problem discussed at the outset was how a third party could verify the associations between trusted authorities arranged in a hierarchy without having to follow a trail of certificates. In fact, the above-described IBC encryption/decryption method offers one possible solution. Consider the situation where a trusted authority at one level in the hierarchy has an IBC public key Q_{ID} / private key S_{ID} pair with the private key being provided by a trusted authority in the next level up on the basis of the ID of the lower-level trusted authority and the private key s of a normal public key *(P, sP)* / private keys pair held by the higher-level trusted authority. A third party could then check that the lower-level trusted authority was associated with the higher level one by an IBC-based challenge/response mechanism. More particularly, the third party could encrypt a nonce (random number) using both the public key element *sP* of the higher-level trusted authority and the IBC public key Q_{ID} of the lower-level trusted authority. The third party sends the encrypted nonce to the lower-level trusted authority and asks it to decrypt and return the nonce - the lower-level trusted authority will only be able to do this if it has (or can get) the key $S_{ID}(= sQ_{ID})$ from the higher-level trusted authority. Thus, if the lower-level trusted authority can return the decrypted nonce, the association between the lower-level trusted authority and the higher level trusted authority is proved. Whilst this approach is viable, it involves an exchange of messages between the third party and the lower-level trusted authority and also (if the lower-level trusted authority does not already have its IBC private key) between the lower-level trusted authority and the higher-level trusted authority. In many situation this may either not be possible or undesirable- for example, the third party may wish to check the association between the trusted authorities offline or the third party may not wish to let it be known that it is carrying out the check.

[0026] It is an object of the present invention to provide a way of checking the association between two parties that obviates at least some of the difficulties noted above.

Summary of the Invention

[0027] According to a first aspect of the present invention, there is provided a method of enabling a third party to verify an association between a first party associated with a first element, of a first algebraic group, and a second party associated with a second element, of a second algebraic group, formed from an identifier string of the second party, wherein:

- there exists a computable bilinear map for the first and second elements;
- the first party has a first secret and computes a first product from the first secret and the first element;
- the second party has both a second secret, and a shared secret provided by the first party as the product of the first secret and the second element;
- the second party computes first, second and third verification parameters as the product of the second secret with said shared secret, the second element and the first element respectively.

[0028] Using the non-secret data elements and a function p providing the bilinear mapping, a third party can verify the existence of an association between first and second parties by:

- computing the second element from the identifier string of the second party;
- carrying out a first check:

  p(third verification parameter, computed second element)
  = p(first element, second verification parameter)
- carries out a second check:

p(first element, first verification parameter)
= p(first product, second verification parameter)
the association between the first and second parties being treated as verified if both checks are passed.

[0029] According to a second aspect of the present invention, there is provided a method of verifying an association between a first party associated with a first element, of a first algebraic group, and a second party associated with a second element, of a second algebraic group; the first and second elements being such that there exists a bilinear mapping p for these elements; the method comprising carrying out the following operations:

- receiving both data indicative of said first element, and a first product formed by the first party from a first secret and the first element;
- receiving in respect of the second party both an identifier string, and first, second and third verification parameters;
- computing the second element from the identifier string of the second party;
- carrying out a first check:
  p(third verification parameter, computed second element)
  = p(first element, second verification parameter)
- carrying out a second check:
  p(first element, first verification parameter)
  = p(first product, second verification parameter)

  the association between the first and second parties being treated as verified if both checks are passed.

[0030] According to a third aspect of the present invention, there is provided apparatus arranged to enable a third party to verify an association between the apparatus and a first party that has a first secret and is associated with a first element, of a first algebraic group; the apparatus being associated with a second element, of a second algebraic group, and the first and second elements being such that there exists a bilinear mapping p for these elements; the apparatus comprising:

- a memory for holding a second secret and an identifier string associated with the apparatus,
- means for forming said second element from said identifier string,
- means for receiving from the first party a shared secret based on said first secret and said first element, and for storing this shared secret in the memory,
- means for computing first, second and third verification parameters as the product of the second secret with said shared secret, said second element and said first element respectively, and
- means for making available said identifier string and said verification parameters to the third party.

[0031] According to a fourth aspect of the present invention, there is provided apparatus for verifying an association between a first party associated with a first element, of a first algebraic group, and a second party associated with a second element, of a second algebraic group; the first and second elements being such that there exists a bilinear mapping p for these elements; the apparatus comprising:

- means for receiving both data indicative of the first element, and a first product formed by the first party from a first secret and the first element:
- means for receiving in respect of the second party both an identifier string, and first, second and third verification parameters;
- means for computing the second element from the identifier string of the second party;
- means for carrying out a first check:
  p(third verification parameter, computed second element)
  = p(first element, second verification parameter)
- means for carrying out a second check:
  p(first element, first verification parameter)
  = p(first product, second verification parameter)
- means responsive to both checks being passed, to confirm that there exists an association between the first and second parties.

[0032] The present invention also encompasses computer program products both for providing verification parameters enabling verification of an association between two parties, and for carrying out a verification check using these parameters.

Brief Description of the Drawings

[0033] Embodiments of the invention will now be described, by way of non-limiting example, with reference to the accompanying diagrammatic drawings, in which:

- **Figure 1** is a diagram showing prior art cryptographic processes based on elliptic curve cryptography using Tate pairings;
- **Figure 2** is a diagram illustrating a first embodiment of the invention illustrating for generalized first and second parties, how a third party can verify an association between first and second parties;
- **Figure 3** is a diagram of a second embodiment involving a hierarchy of a first-level trusted authority and a second-level trusted authority; and
- **Figure 4** is a diagram of a third embodiment involving an n-level hierarchy of trusted authorities.

Best Mode of Carrying Out the Invention

[0034] Considering first the situation where there is an association between a first party and a second party which the second party would like to be able to prove to a third party; the nature of the association concerned is not relevant to the present discussion but could, for example, be a trust relationship (e.g. the second party is trusted to act on behalf of the first party in respect of certain matters) or simply a biological relationship (e.g. the first party is a parent and the second is a child of the first party).

[0035] In order to enable the second party to prove this association, the first party provides the second party with a secret, herein referred to as a "shared secret", though there is no requirement on the first party to keep a copy of this shared secret after giving it to the second party. The nature of the shared secret is such that it enables the second party to prove its association with the first party without giving away the shared secret.

[0036] According to the present invention, the above-described arrangement is enabled by the use of bilinear mappings as will now be explained with reference to embodiments based on modified Tate pairings (though, of course, other pairings such as modified Weil pairings can alternatively be used). The notations and definitions given in the introductory portion of the present specification also apply to what follows.

[0037] The first party has its own secret s_1 and an associated point P on an elliptic curve. The first party makes P and the combination $s_1P(=R)$ publicly available in any suitable manner. The second party also has its own secret s_2 and an associated point Q on the same elliptic curve as P. The second party makes Q and the combination s_2Q publicly available in any suitable manner. It will be appreciated that reference to an element being made publicly available simply means making it available to third parties who have an interest and right to know the element and does not necessarily imply unrestricted distribution.

[0038] The second party is provided with s_1Q by the first party as the shared secret that is to be used in establishing to the third party the association between the second party and the first party. In order to keep the shared secret s_1Q secret whilst providing the third party with the information it needs to verify the association between the first and second parties, the second party combines s_1Q with s_2 and makes the resulting combination s_1s_2Q public.

[0039] Recapping so far, the elements associated with the first and second parties are:

First party:

Secret data: s_1
Public data: P, $R(=s_1P)$

Second party:

Secret Data: s_2, s_1Q
Public data: Q, s_1s_2Q, s_2Q

[0040] It is assumed that the third party reliably knows P and $R(=s_1P)$, the public data of the first party. The third party has also received, in respect of the second party: the point Q; an element, herein called X, that is purportedlys s_1s_2Q; and an element, herein called Y, that is purportedly s_2Q. In order to check whether X truly does contain s_1, the third party checks the following :

$$t(P, X) = t(R, Y) \qquad \text{Test 1}$$

[0041] Because $R=s_1P$, the above will only be valid if X is equal to s_1Y. This would prove that the second party must have a shared secret containing s_1 which only it and the first party know (thus proving the association between the parties) were it not for the possibility that, since s_1P is public, the second party could have constructed Q as mP, where $m \in F_q$, and then used m, s_2 and s_1P to construct X as s_1s_2mP and Y as s_2mP. In other words, if the second party can construct its Q from P then, it can pass Test 1 without needing to ask for a shared secret from the first party.

[0042] It is therefore necessary for the third party to be satisfied that Q has not been formed by multiplying P by m (it being appreciated that because the discrete logarithm problem is hard, the third party cannot discover if Q of the form mP- though, of course, if $m=1$, this will be apparent). To this end, the point Q is required to be derived from an identifier string ID using the map-to-point hash function H_1 because in this case even if Q happened to be equal to mP (which is highly unlikely), the second party would neither be aware of this nor able to separate out m and use it to generate an X of the form s_1s_2mP. It is not, of course, possible for the second party to work backwards from a value of m to produce the string ID that would give rise to m using the map-to-point function.

[0043] To emphasise the fact that Q originates from an identifier, it is suffixed with "ID" in the following discussion; thus:

$$Q_{ID} = H_1(ID)$$

where the identifier string ID can be any string and typically, though not necessarily, serves to identify the second party in plain language.

[0044] So now if the second party makes public the string ID rather than (or in addition to) Q_{ID}, the third party can use the string ID to form the point Q_{ID} thereby re-assuring itself that the second party has not used a value m to form Q as mP. However, the third party also needs to be able to link this legitimate Q_{ID} to the elements used in Test 1 - in particular, the third party needs to be sure that the element Y contains the legitimate Q_{ID} derived from ID. To this end, the third party must carry out a second test for which purpose the second party must provide a further quantity, herein called Z, that is purportedly equal to s_2P. The second test is of the following form:

$$t(Z, Q_{ID}) = t(P, Y) \qquad \text{Test 2}$$

[0045] If this is true, then the second party knows that Y must contain Q_{ID}.

[0046] The above test (Test 1) is now therefore adequate to prove that the second party does indeed have a shared secret of the form s_1Q_{ID} which must have been provided by the first party, thereby proving there is an association between the first and second parties.

[0047] Recapping, and as shown in Figure 2, the elements associated with the first and second parties 5, 6 are:

First party 5:

Secret data: s_1
Public data: P, $R=s_1P$

Second party 6 :

Secret data: s_2,
Public data: ID, $X=s_1s_2Q_{ID}$, $Y=s_2Q_{ID}$, $Z= s_2P$

and the third party 7 carries out the following:

Q_{ID} = map-to-point H_1(ID);
Test 2;
Test 1.

[0048] The requirements for the third party to be able to verify the association between the first and second parties (respectively higher-level and lower-level parties in the association hierarchy) can thus be expressed as follows:

- the first party must have a public key (P, R)/private key s_1 key pair where $R=s_1P$; it may be noted that P could be based on an identity string for the first party by using the map-to-point hash H_1.
- the second party must have an IBC public key ID / private *keys* s_1Q_{ID} key pair where $Q_{ID}=H_1$(ID).

- using a secret s_2 the second party must form three public verification parameters (X, Y, Z) by multiplying by s_2:
  - the point P that is part of the public key of the first party,
  - the point Q_{ID} of the second party,
  - the private part $s_1 Q_{ID}$ of the second party's IBC key pair.

[0049] In applying the two Tests 1 and 2, the point P is the point that is part of the public key of the first (higher-level) party, the other part of the key being R, whilst the point Q_{ID} is the point derived from the identity of the second (lower-level) party using the map-to-point hash function H_1 and the parameters X, Y and Z are all supplied by the second party.

[0050] Other ways of characterising the parameters referred to above as the "verification parameters" are also possible; for example, it may be noted that two of these parameters, namely $Y(=s_2 Q_{ID})$ and $Z(=s_2 P)$ can each be viewed as part of the public key of a respective standard public/private key pair that involves the point concerned and has a private key of s_2.

[0051] Figure 3 illustrates the application of the foregoing to an hierarchical arrangement of two trusted authorities 60 and 70 where the latter has issued a user 80 with an IBC private key.

[0052] More particularly, Figure 3 shows a first computer entity 10, a second computer entity 20, a third computer entity 30 and a fourth computer entity 40 connected via a network 50, for example the Internet. The first computer entity 10 represents a first trusted authority 60, for example a company, the second computer entity 20 represents a second trusted authority 70, for example a division within the company and the third computer entity 30 represents a user 80, for example a worker within the company. The fourth computer entity 40 represents, for example, a business partner 90 of the company that wishes to interact with the user 80.

[0053] The first, second, third and fourth computer entities 10, 20, 30, 40 are conventional program-controlled computing devices though specialised hardware may be provided to effect particular cryptographic processes.

[0054] The first computer entity 10 and second computer entity 20 form a trusted authority hierarchy in which the first computer entity 10 acts as a root, or first level, trusted authority 60 and the second computer entity 20 acts as a second level trusted authority 70. The first-level trusted authority 60 has a standard public key (P, R_{TA1}) / private keys key pair where $R_{TA1}= s_1 P$. The second-level trusted authority 20 has an IBC public/private key pair the private key S_{TA2} of which has been generated by the first-level trusted authority 60 using its private key s_1 and Q_{TA2}, where $Q_{TA2}=H_1$ (TA2) and "TA2" is an identity string associated with the second-level trusted authority 70. Table 1 sets out the keys held by the first-level and second-level trusted authorities 60 and 70.

Table 1

Entity	Standard Private Key	Standard Public key	ID Based Private Key	ID Based Pubic key
First-level TA	s_1	P, $R_{TA1}(=s_1 P)$		
Second-level TA			$S_{TA2}=s_1 Q_{TA2}$	$Q_{TA2}= H_1(TA2)$

[0055] Once in the possession of the IBC private key S_{TA2} (the "master private key") the second-level trusted authority 70 is able to produce a set of verification parameters X, Y and Z enabling a third party to verify, without further interaction with the first-level trusted authority and without the need for digital certificates, that the private key of the IBC public/private key pair of the second-level trusted authority 70 could only have been generated by the first-level trusted authority 60. More particularly, the second-level trusted authority 70 selects a random number r where $r \in F_q$; the random number r is a "pseudo-master private key". Once the pseudo-master key has been selected the second-level trusted authority 70 generates the following public verification parameters:

$$rs_1 Q_{TA2},\ rQ_{TA2} \text{ and } rP$$

that respectively correspond to the parameters X, Y and Z of the above-described Tests 1 and 2.

[0056] It should be noted that even though in the above example the second-level trusted authority 70 has created a single pseudo-master private key, the second-level trusted authority 70 could generate any number of pseudo-master private keys.

[0057] It may also be noted that the second-level trusted authority 70 is likely also to have one or more standard public/private key pairs. For example, the pseudo-master private key r could be used as the private key and combined either with P or Q_{ID} or another point in G_1 not computed from an existing point, to form a corresponding public key. Alternatively, a completely separate private key s_2 could be generated where $s_2 \in F_q$ and used with P or Q_{ID} or another point in G_1 not computed from an existing point, to form a corresponding public key.

[0058] The user 80 registers with the second trusted authority 70 to obtain an associated IBC private key for the user's public key, where the user's public key could be any form of identifier, for example the user's name 'Bob', and the map-to-point hash H_1(Bob) of this identifier maps to a point Q_{Bob} in G_1. The IBC private key provided to the user 80 is a combination of the user's public key and the second-level trusted authority's pseudo private key i.e. the user's private key is rQ_{Bob}.

[0059] To send an encrypted message to the user 80, the third-party business partner 90 can now use the IBC public key of the user 80 and the public key of the second-level trusted authority 70 used by user 80; in doing this, the third party 90 can be sure that the user will only be able to decrypt the message if the user is known to the second-level trusted authority 70 since the IBC private key needed for decryption must be provided by that authority.

[0060] The third party 90 can also verify that the second-level trusted authority 70 (company division) is associated with the first-level trusted authority (company). To do this, the third party 90 uses the identity "TA2" and public verification parameters rs_1Q_{TA2}, rQ_{TA2} and rP of the second-level trusted authority 70, together with the public key P, $R_{TA1}(=_{S1}P)$ of the first-level trusted authority 60, to carry out the Tests 1 and 2 described above with respect to Figure 2. More particularly:

- the third party 90 first forms Q_{TA2} from the identity string "TA2" using the map-to-point hash function H_1;
- the third party 90 carries out Test 2 by checking

$$t(Z, Q_{TA2}) = t(P, Y)$$

where $Z= rP$ and $Y= rQ_{TA2}$ and Q_{TA2} is the element just formed from the identity "TA2"; this check, if passed, confirms that the element Y contains Q_{TA2}

- the third party 90 carries out Test 1 by checking

$$t(P, X) = t(R_{TA1}, Y)$$

where $R_{TAI}=s_1P$ and $X= rs_1Q_{TA2}$; this check, if passed, confirms that X must contain s_1 which the second-level trusted authority 70 must have obtained in a non-public element from the first-level trusted authority 60.

[0061] Of course, because the second-level trusted authority has published its point Q_{TA2} (or the underlying identifier "TA2") as well as the element rQ_{TA2} thereby providing a standard public/private key pair, it would be possible for the user 80 itself to produce a set of verification parameters to enable the third party 90 to verify the existence of an association between the user 80 and the second-level trusted authority 70 without needing to send a message to the user. To produce the required verification parameters the user 80 picks a random number r_B where $r_B \in F_q$ and generates the parameters:

$$r_B rQ_{Bob},\ r_B Q_{Bob} \text{ and } r_B Q_{TA2}$$

respectively corresponding to the parameters X, Y and Z. In this case, in the Tests 1 and 2, the element P is, of course, replaced by Q_{TA2} and the element R by rQ_{TA2} as Q_{TA2} is now the point associated with the higher-level party. In fact, where the second-level trusted authority has provided one or more other standard public/private key pairs, the public values of any such pair can be used for the elements P and R in the previously stated forms of the Tests.

[0062] Figure 4 of the accompanying drawings illustrates for an n-level hierarchy of trusted authorities TA1 to TAn, a possible organisation of keys and verification parameters. In this example, each trusted authority such as authority TAi (where $1<i<=n$) has:

- a standard public/private key pair, the private key of this key pair being a secret S_i and the public key being (P_i, S_iPi) where $P_i=H_1("TAi")$ that is, the map-to-point hash of the identity of the authority;
- an IBC key pair, the public key of this key pair being the identity TAi of the trusted authority and the secret key being the product of the map-to-point hash of this identity and the secret s_{i-1} of the next level up trusted authority;
- two additional verification parameters $s_is_{i-1}P_i$ and s_iP_{i-1} (corresponding to X and Z above, the verification parameter $Y= s_iP_i$ already being present in the public key of the standard key pair).

[0063] The root trusted authority TA1 simply has a standard public key(P_1, s_1P)/private key s_1 key pair.

[0064] With this hierarchy, it is possible to verify the association between each parent/child pairing of trusted author-

ities in the hierarchy thereby enabling a check to be made that any non-root trusted authority, from the lowest level (or leaf) authority upwards, is associated with the root trusted authority.

[0065] It will be appreciated that many variants are possible to the above described embodiments of the invention.

Claims

1. A method of enabling a third party (90) to verify an association between a first party (60) associated with a first element (P), of a first algebraic group (G_1), and a second party (70) associated with a second element (Q_{TA2}), of a second algebraic group (G_1), formed from an identifier string ("TA2") of the second party, wherein:

   - there exists a computable bilinear map for the first and second elements (P,Q_{TA2});
   - the first party has a first secret (s_1) and computes a first product (R_{TA1}) from the first secret (s_1) and the first element (P);
   - the second party has both a second secret (r), and a shared secret ($s_1 Q_{TA2}$) provided by the first party as the product of the first secret (s_1) and the second element (Q_{TA2});
   - the second party computes first (X), second (Y) and third (Z) verification parameters as the product of the second secret (r) with said shared secret ($s_1 Q_{TA2}$), said second element (Q_{TA2}) and said first element (P) respectively.

2. A method according to claim 1, wherein the second party (70) generates a further shared secret (rQ_{Bob}) from the second secret (r) and an identifier string ("Bob") of a fourth party (80), the second party (70) passing this further shared secret to the fourth party (80) for use by the latter as the private key of a public/private key pair the public key of which is formed by the identifier string ("Bob") of the fourth party (80).

3. A method according to claim 1 or claim 2, wherein the first and second parties are respectively parent and child trusted authorities (60,70) in a hierarchy of trusted authorities.

4. A method according to any one of the preceding claims, wherein the first and second algebraic groups (G_1, G_1) are the same.

5. A method according to any one of the preceding claims, wherein the first and second elements (P,Q_{TA2}) are points on the same elliptic curve.

6. A method of verifying an association between the first and second parties (60,70) of claim 1 by using a function p providing said bilinear map; the method comprising carrying out the following operations using the non-secret data elements of claim 1:

   - computing said second element (Q_{TA2}) from the identifier string ("TA2") of the second party;
   - carrying out a first check:
     p(third verification parameter (Z), computed second element (Q_{TA2}))
     = p(first element (P), second verification parameter (Y))
   - carrying out a second check:
     p(first element (P), first verification parameter (X))
     = p(first product (R_{TA1}), second verification parameter (Y))

   the association between the first and second parties (60,70) being treated as verified if both checks are passed.

7. A method according to claim 8, wherein said bilinear mapping function p is based on a Tate or Weil pairing.

8. A method of verifying an association between a first party (60) associated with a first element (P), of a first algebraic group (G_1), and a second party (70) associated with a second element, of a second algebraic group (G_1); the first and second elements (P,Q_{TA2})being such that there exists a bilinear mapping p for these elements; the method comprising carrying out the following operations:

   - receiving both data indicative of the first element (P) and a first product (R_{TA1}) formed by the first party from a first secret (s_1) and the first element (P);
   - receiving in respect of the second party (70) both an identifier string ("TA2"), and first (X), second (Y) and third (Z) verification parameters;

- computing the second element (Q_{TA2}) from the identifier string ("TA2") of the second party (70);
- carrying out a first check:
  p(third verification parameter (Z), computed second element (Q_{TA2}))
  = $\hat{p}$(first element (P), second verification parameter (Y))
- carrying out a second check:
  p(first element (P), first verification parameter (X))
  = $\hat{p}$(first product (R_{TA1}), second verification parameter (Y))

  the association between the first and second parties (60,70) being treated as verified if both checks are passed.

9. A method according to claim 8, wherein said bilinear mapping p is based on a Tate or Weil pairing.

10. A method according to claim 8 or claim 9, wherein the first and second algebraic groups (G_1, G_1), are the same.

11. A method according to any one of claims 8 to 10, wherein the first and second elements (P, Q_{TA2}) are points on the same elliptic curve.

12. Apparatus arranged to enable a third party (90) to verify an association between the apparatus (20) and a first party (60) that has a first secret (s_1) and is associated with a first element (P), of a first algebraic group (G_1); the apparatus (20) being associated with a second element (Q_{TA2}), of second algebraic group(G_1), and the first and second elements (P, Q_{TA2}) being such that there exists a bilinear mapping p for these elements; the apparatus (20) comprising:

- a memory for holding a second secret (r) and an identifier string ("TA2") associated with the apparatus,
- means for forming the second element (Q_{TA2}) from said identifier string ("TA2"),
- means for receiving from the first party (60) a shared secret ($s_1 Q_{TA2}$) based on said first secret (s_1) and said second element (Q_{TA2}), and for storing this shared secret in the memory,
- means for computing first (X), second (Y) and third (Z) verification parameters as the product of the second secret (r) with said shared secret ($s_1 Q_{TA2}$), said second element (Q_{TA2}) and said first element (P) respectively, and
- means for making available said identifier string ("TA2") and said verification parameters (X, Y,Z) to the third party (90).

13. Apparatus according to claim 12, wherein the first and second algebraic groups (G_1, G_1) are the same.

14. Apparatus according to claim 12 or 13, wherein the first and second elements (P, Q_{TA2}) are points on the same elliptic curve.

15. Apparatus for verifying an association between a first party (60) associated with a first element (P), of a first algebraic group (G_1), and a second party (70) associated with a second element (Q_{TA2}), of a second algebraic group (G_1); the first and second elements (P, Q_{TA2}) being such that there exists a bilinear mapping p for these elements; the apparatus comprising:

- means for receiving both data indicative of the first element (P) and a first product (R_{TA1}) formed by the first party from a first secret (s_1) and the first element (P):
- means for receiving in respect of the second party (70) both an identifier string ("TA2"), and first, second and third verification parameters (X, Y, Z);
- means for computing the second element (Q_{TA2}) from the identifier string ("TA2") of the second party (70);
- means for carrying out a first check:
  p(third verification parameter (Z), computed second element (Q_{TA2}))
  = $\hat{p}$(first element (P), second verification parameter (Y))
- means for carrying out a second check:
  p(first element (P), first verification parameter (X))
  = $\hat{p}$(first product (R_{TA1}), second verification parameter (Y))
- means responsive to both checks being passed, to confirm that there exists an association between the first (60) and second (70) parties.

16. Apparatus according to claim 15, wherein said bilinear mapping p is based on a Tate or Weil pairing.

17. Apparatus according to claim 15 or claim 16, wherein the first and second elements (P, Q_{TA2}) are points on the same elliptic curve.

18. A computer program product arranged, when installed in computing apparatus, to condition the apparatus to be of the form set out in claim 12.

19. A computer program product arranged, when installed in computing apparatus, to condition the apparatus to be of the form set out in claim 15.

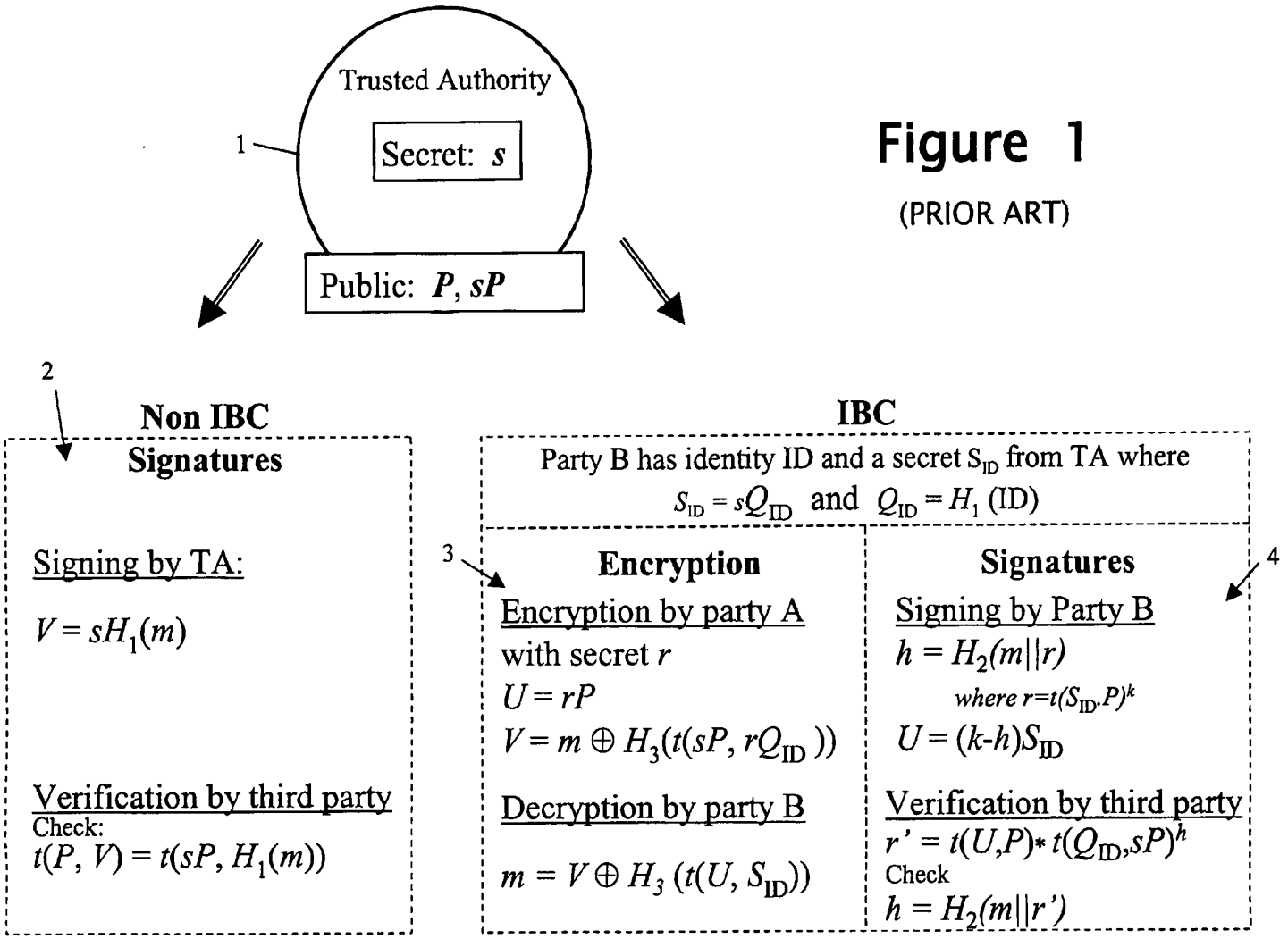

Figure 1

(PRIOR ART)

EP 1 378 821 A2

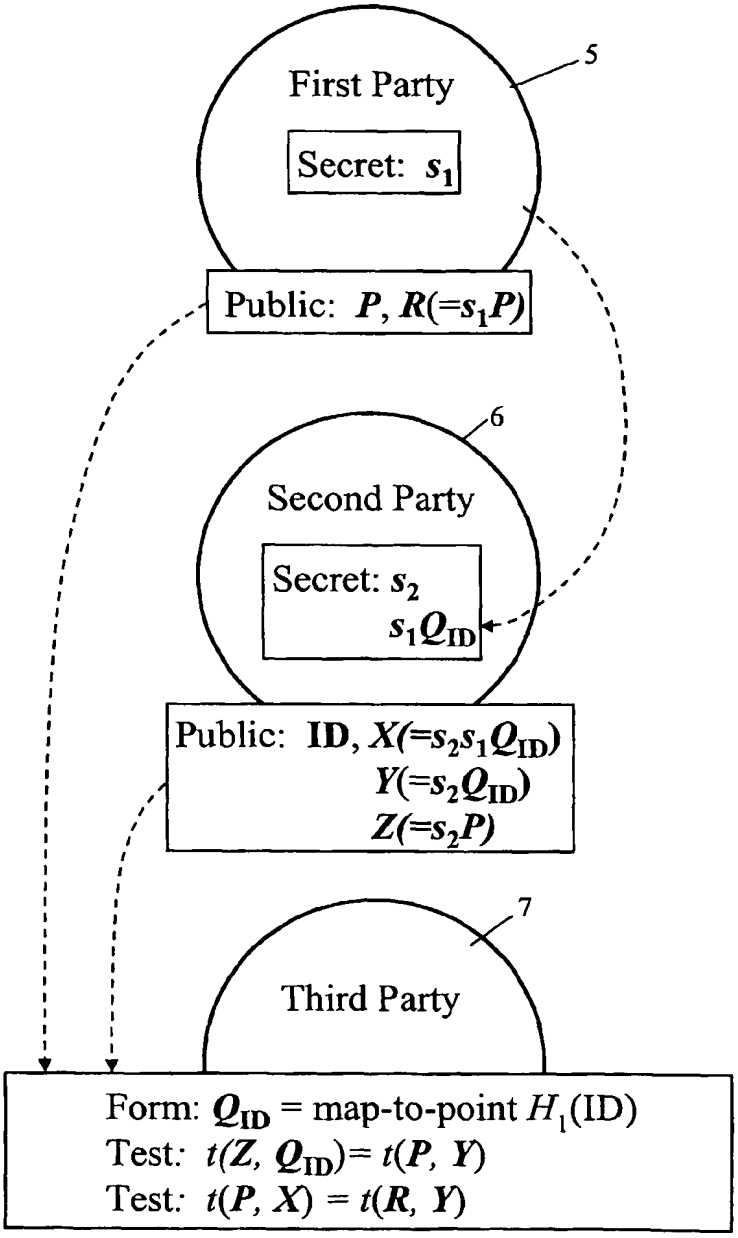
First Party
5
Secret: s_1
Public: P, $R(=s_1P)$
6
Second Party
Secret: s_2
s_1Q_{ID}
Public: ID, $X(=s_2s_1Q_{ID})$
$Y(=s_2Q_{ID})$
$Z(=s_2P)$
7
Third Party
Form: Q_{ID} = map-to-point H_1(ID)
Test: $t(Z, Q_{ID}) = t(P, Y)$
Test: $t(P, X) = t(R, Y)$

Figure 2

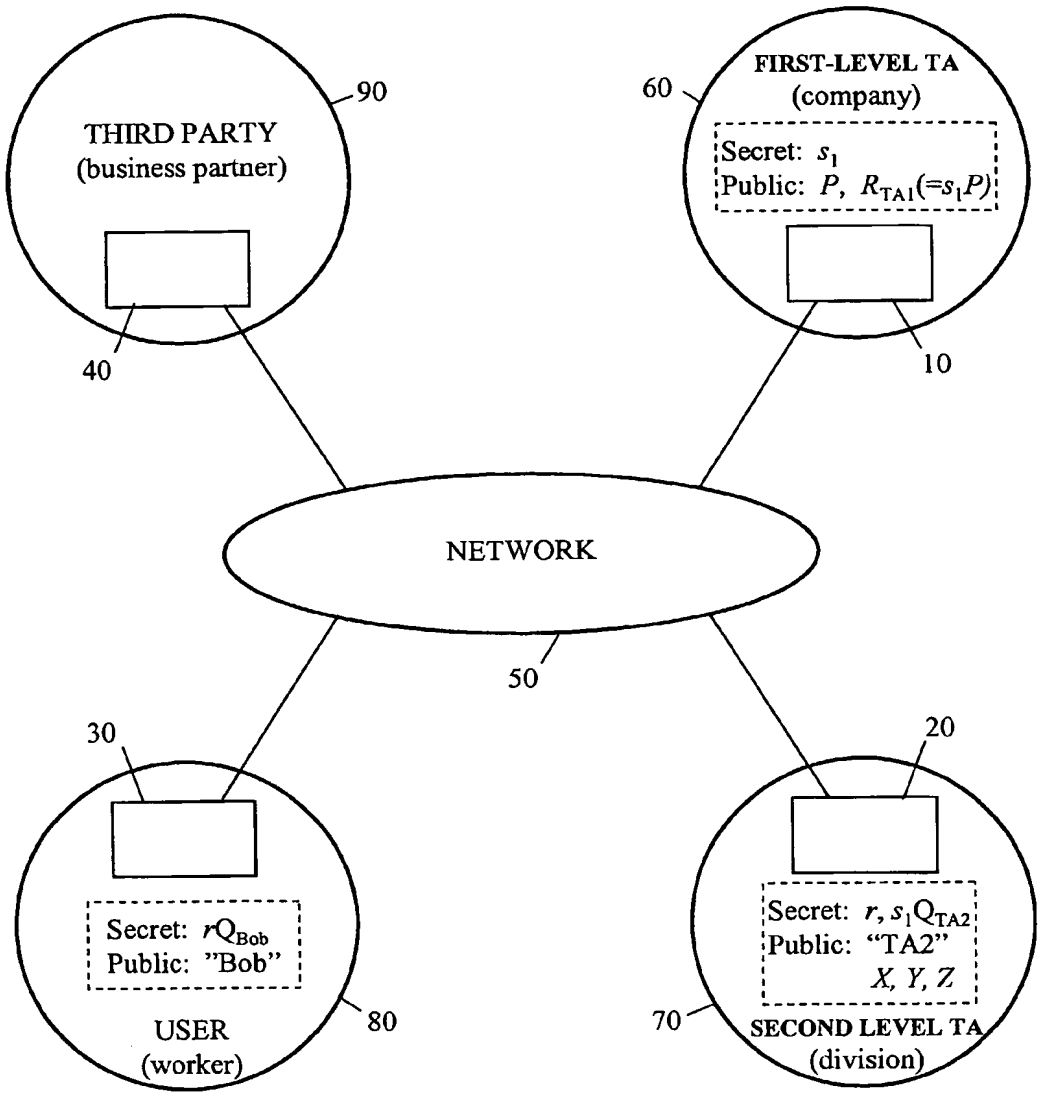
THIRD PARTY
(business partner)
90
40
FIRST-LEVEL TA
(company)
60
Secret: s_1
Public: P, $R_{TA1}(=s_1P)$
10
NETWORK
50
30
Secret: rQ_{Bob}
Public: "Bob"
USER
(worker)
80
20
Secret: r, s_1Q_{TA2}
Public: "TA2"
X, Y, Z
70
SECOND LEVEL TA
(division)

Figure 3

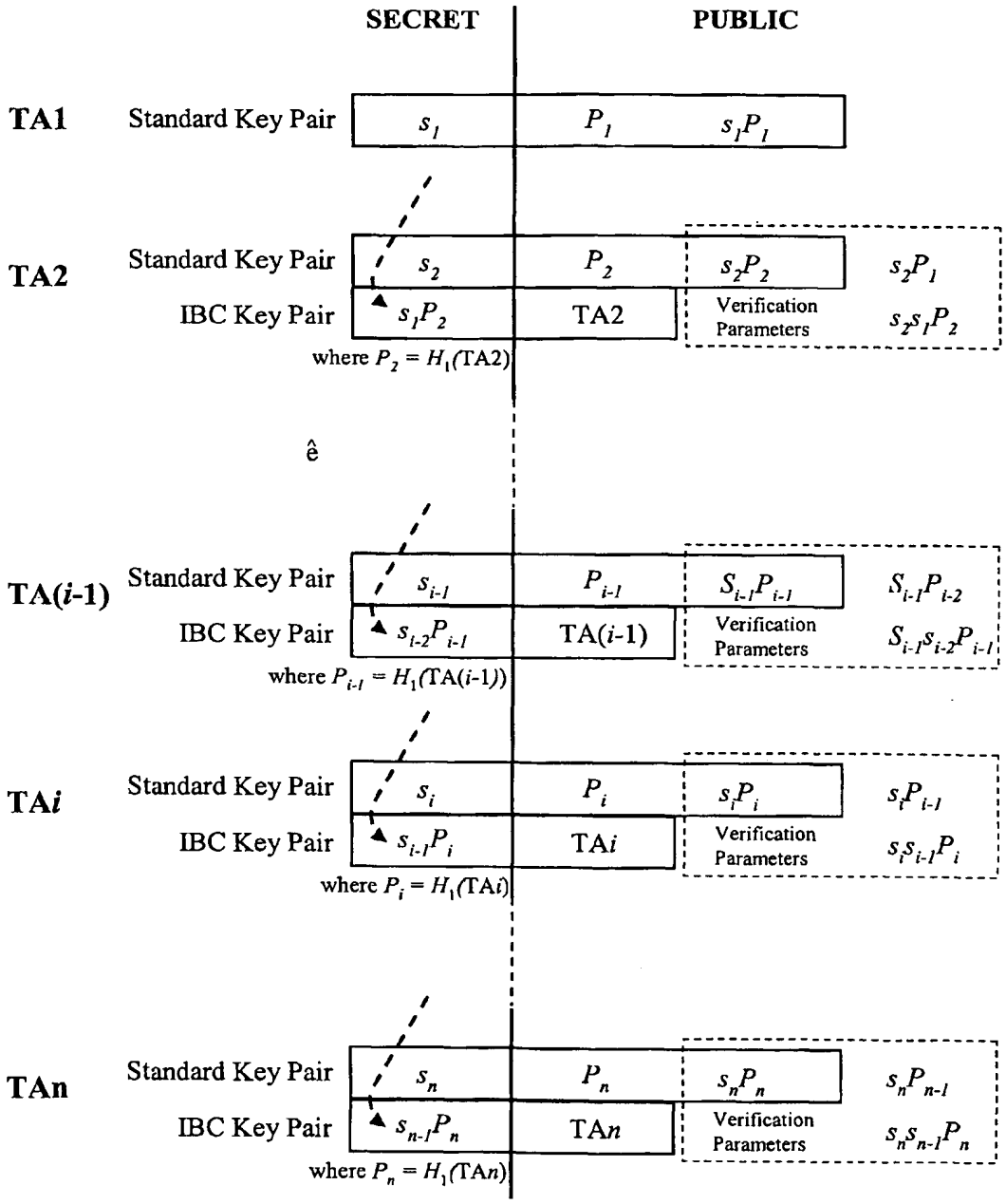
SECRET
PUBLIC
TA1
Standard Key Pair
s_1
P_1
s_1P_1
TA2
Standard Key Pair
s_2
P_2
s_2P_2
s_2P_1
IBC Key Pair
s_1P_2
TA2
Verification Parameters
$s_2s_1P_2$
where $P_2 = H_1(TA2)$
ê
TA(i-1)
Standard Key Pair
s_{i-1}
P_{i-1}
$S_{i-1}P_{i-1}$
$S_{i-1}P_{i-2}$
IBC Key Pair
$s_{i-2}P_{i-1}$
TA(i-1)
Verification Parameters
$S_{i-1}s_{i-2}P_{i-1}$
where $P_{i-1} = H_1(TA(i-1))$
TAi
Standard Key Pair
s_i
P_i
s_iP_i
s_iP_{i-1}
IBC Key Pair
$s_{i-1}P_i$
TAi
Verification Parameters
$s_is_{i-1}P_i$
where $P_i = H_1(TAi)$
TAn
Standard Key Pair
s_n
P_n
s_nP_n
s_nP_{n-1}
IBC Key Pair
$s_{n-1}P_n$
TAn
Verification Parameters
$s_ns_{n-1}P_n$
where $P_n = H_1(TAn)$

Figure 4